Sun Java System Message Queue 4.2 リリースノート

Sun Microsystems, Inc.
4150 Network Circle
Santa Clara, CA 95054
U.S.A.

Part No: 820–5641
2008 年 7 月

080819@20795

目次

◆ ◆ ◆ 第 1 章

Sun Java System Message Queue 4.2 リリースノート

Version 4.2

Part No. 820-5641

このリリースノートには、Sun Java™ System Message Queue 4.2 のリリース時点で得られる重要な情報が含まれています。ここでは、新機能、拡張機能、既知の問題、制限事項などについて説明します。Message Queue 4.2 をご使用になる前に、このリリースノートをお読みください。

このリリースノートには、4.0 および 4.1 リリースの Message Queue に関する情報も含まれています。4.0 および 4.1 リリースで導入された機能については、それぞれ、22 ページの「Message Queue 4.0 の新機能」および 18 ページの「Message Queue 4.1 の新機能」を参照してください。

これらのリリースノートの最新版は、Sun Java System Message Queue ドキュメント Web サイト `http://docs.sun.com/coll/1307.3` から入手できます。ソフトウェアをインストールおよび設定する前だけでなく、それ以降も定期的にこの Web サイトをチェックして、最新のリリースノートと製品マニュアルを確認してください。

このリリースノートは、次の節で構成されています。

- 53 ページの「このマニュアルに関するコメント」
- 54 ページの「その他の情報」

このマニュアル内で参照している第三者の URL は、追加の関連情報を提供します。

このマニュアル内で引用する第三者の Web サイトの可用性について Sun は責任を負いません。こうしたサイトやリソース上の、またはこれらを通じて利用可能な、コンテンツ、広告、製品、その他の素材について、Sun は推奨しているわけではなく、Sun はいかなる責任も負いません。こうしたサイトやリソース上の、またはこれらを経由して利用可能な、コンテンツ、製品、サービスを利用または信頼したことに伴って発生した (あるいは発生したと主張される) いかなる損害や損失についても、Sun は一切の責任を負いません。

リリースノートの変更履歴

次の表は、Message Queue 製品のすべての 4.x リリースの日付と、各リリースに関連するこのマニュアルの変更内容を示しています。

表 1–1　変更履歴

日付	変更点
2006 年 5 月	Message Queue 4.0 のこのマニュアルの初回リリース。
2007 年 1 月	Message Queue 4.1 Beta のこのマニュアルの初回リリース。JAAS サポートの説明が追加されています。
2007 年 4 月	Message Queue 4.1 Beta のこのマニュアルの 2 番目のリリース。高可用性の機能が追加されています。
2007 年 9 月	Message Queue 4.1 のこのマニュアルの 3 番目のリリース。Java Enterprise System Monitoring Framework のサポート、固定 C ポート、バグ修正、およびその他の機能に関する説明が追加されています。
2008 年 4 月	Message Queue 4.2 のこのマニュアルの初回ドラフトリリース。このリリースの新機能が追加されています。

Message Queue 4.2 のインストールまたはアップグレード

Message Queue 4.2 インストーラを使用して、Message Queue 4.2 の新規インストールまたは Message Queue 3.6 以降からのアップグレードを実行できます。Solaris、Linux、および Windows プラットフォームでのインストールまたはアップグレードに関する手順およびその他すべての情報は、『Sun Java System Message Queue 4.2 Installation Guide』に記載されています。このマニュアルは、Message Queue 4.2 に関する更新は行われていません。

3.6 より前のバージョンの Message Queue からアップグレードする場合は、『*Sun Java Enterprise System 5* アップグレードガイド *(UNIX* 版*)*』、『Sun Java Enterprise System 5 Update 1 Upgrade Guide for UNIX 』を参照してください。

さらに、40 ページの「インストールに関する情報」で、インストールとアップグレードに関する既知の問題および制限事項についても確認してください。

Message Queue 4.2 でサポートされるプラットフォームとコンポーネント

この節は、Message Queue 4.2 システム要件に関する次のトピックで構成されています。

オペレーティングシステムプラットフォーム のサポート

Message Queue 4.2 は、Solaris、Linux、および Windows オペレーティングシステムのプラットフォームでサポートされます。表 1–2 に、サポートされる各プラットフォームのバージョンを示します。各プラットフォームのハードウェア要件については、『Sun Java System Message Queue 4.2 Installation Guide』を参照してください。

表 1–2　サポートされるプラットフォームのバージョン

プラットフォーム	サポートされるバージョン
Solaris	Solaris 9 (SunOS 5.9)、すべての更新版 (SPARC、 *x*86) Solaris 10 (SunOS 5.10)、すべての更新版 (SPARC、 *x*86、*x*64)

表 1–2　サポートされるプラットフォームのバージョン　　(続き)

プラットフォーム	サポートされるバージョン
Linux	Red Hat Enterprise Linux Advanced Server 3.0、4.0、5.0、すべての更新版、32 および 64 ビットバージョン (*x*86、*x*64)
	Red Hat Enterprise Linux Enterprise Server 3.0、4.0、5.0、すべての更新版、32 および 64 ビットバージョン (*x*86、*x*64)
Windows	Windows Vista
	Windows XP Professional、SP2 (*x*86)[1]
	Windows 2000 Advanced Server、SP4 (*x*86)[2]
	Windows Server 2003 Standard Edition、Enterprise Edition、SP2、32 および 64 ビットバージョン (*x*86、*x*64)[3]

[1] Home Edition、Tablet PC Edition、Media Center Edition はサポートされません

[2] Professional Edition、Server Edition はサポートされません

[3] Web Edition、Small Business Server Edition はサポートされません

システムの仮想化のサポート

システムの仮想化は、複数のオペレーティングシステム (OS) インスタンスを共用ハードウェア上で個別に実行できるようにするテクノロジです。機能的にいうと、仮想化された環境でホストされる OS に配備されたソフトウェアは、通常はベースとなるプラットフォームが仮想化されていることを認識しません。Sun では、精選されたシステムの仮想化と OS の組み合わせについて、その Sun Java System 製品のテストを行っています。これは、Sun Java System 製品が、適切な規模と構成の仮想化された環境で、仮想化されていないシステム上の場合と同様に引き続き機能することを実証するためのテストです。仮想化された環境での Sun Java System 製品に対する Sun のサポートの詳細は、http://docs.sun.com/doc/820-4651 を参照してください。

コンポーネントの依存関係

プラットフォーム固有の要件のほかに、Message Queue 4.2 は特定の基本コンポーネントにも依存します。Message Queue クライアントを開発して実行する場合は、それらのコンポーネントをインストールしてください。表 1–3 に、それらのコンポーネントを示します。ほかのバージョンまたはベンダー製品も実装できますが、それらは Sun Microsystems によるテストが実施されていないため、公式にはサポートされません。

注 – Message Queue インストーラでは、既存の JDK/JRE を選択するか、または JDK Version 1.5.0_15 をインストールできます。

表 1–3　必須のサポートコンポーネント

コンポーネント	サポート対象	サポートされるバージョン
Java Runtime Environment (JRE)	Message Queue ブローカおよび管理ツール	J2SE™ Runtime Environment 1.5.0_15 以降 Java™ SE Runtime Environment 1.6.0 (Sun Microsystems バージョンのみ)
Java Software Development Kit (JDK), Standard Edition	Java クライアントの開発および配備	J2SE Development Kit 1.5.0_15 以降 Java SE Development Kit 1.6.0 (Sun Microsystems 製品バージョンのみ)

表 1–4 に、Message Queue クライアントのサポートを追加するためにインストールできるその他のコンポーネントを示します。示されているすべてのコンポーネントをインストールする必要はありません。たとえば、C クライアントを作成しない場合は、C コンパイラ、C++ 実行時ライブラリ、NSPR、NSS などは必要ありません。

表 1–4　任意のサポートコンポーネント

コンポーネント	サポート対象	サポートされるバージョン
アプリケーションサーバー	HTTP/HTTPS	Sun Java System Application Server Enterprise Edition, Version 9.1 Update Release 2
Web サーバー	HTTP/HTTPS	Sun Java System Web Server Enterprise Edition, Version 7.0, Update 2
データベース	JDBC ベースのデータストア	HADB, Version 4.4.3.5 Java DB (Apache Derby), Version 10.2.2 MySQL Community Edition, Version 5.0 Oracle10g postgreSQL, Version 8.1 注 – PointBase データベースは、現在はサポートされていません。
高可用性データベース	高可用性ブローカクラスタ	HADB, Version 4.4.3.5 MySQL Cluster Edition, Version 5.0 Oracle10g
Lightweight Directory Access Protocol (LDAP) ディレクトリサーバー	Message Queue ユーザーリポジトリおよび管理対象オブジェクト	Sun Java System Directory Server, Version 6.0

表 1–4 任意のサポートコンポーネント (続き)

コンポーネント	サポート対象	サポートされるバージョン
Java Naming and Directory Interface (JNDI)	管理対象オブジェクトのサポートと LDAP ユーザーリポジトリ	JNDI Version 1.2.1 LDAP Service Provider, Version 1.2.2 File System Service Provider, Version 1.2 Beta 3[1]
C 言語のコンパイラおよび同等の C++ ランタイムライブラリ	Message Queue C クライアント	**Solaris:**Sun Studio, Version 11 以降、C++ コンパイラ (標準モード) および C コンパイラ **Linux:** gcc/g++, Version 3.2.3 **Windows:** Microsoft Windows Visual C++, Version 6.0 SP3
Netscape Portable Runtime (NSPR)	Message Queue C クライアント	Version 4.7–1[2]
Network Security Services (NSS)	Message Queue C クライアント	Version 3.11.9–1[2]

[1] 管理対象オブジェクトのサポートのみ。開発とテストでサポートされますが、本稼働環境での開発ではサポートされません

[2] ダウンロードバンドルに共有パッケージとしてバンドルされています

Message Queue 4.2 の新機能および最近のリリース

次の各節では、Message Queue 4.2、4.1、および 4.0 の新機能について説明します。

- 10 ページの「Message Queue 4.2 の新機能」
- 18 ページの「Message Queue 4.1 の新機能」
- 22 ページの「Message Queue 4.0 の新機能」

Message Queue 4.2 の新機能

Sun Java System Message Queue は、多くの機能を備えるメッセージサービスで、Java Messaging Specification (JMS) 1.1 に準拠する信頼性の高い非同期メッセージングを提供します。Message Queue では、JMS 仕様を超える機能も用意され大規模企業のシステム配備のニーズにも対応できるようになっています。

Message Queue 4.2 は、いくつかの機能拡張とバグ修正が施されたマイナーリリースです。この節では、Message Queue 4.2 のインストールまたは Message Queue 4.2 へのアップグレードの方法、およびこのリリースに含まれる新機能について説明します。

- 11 ページの「パブリッシャーまたはサブスクライバの複数の送信先」
- 13 ページの「XML ペイロードメッセージのスキーマ検証」
- 14 ページの「分散トランザクションの C-API サポート」

- 16 ページの「インストーラでの Sun Connection 登録のサポート」
- 18 ページの「MySQL データベースのサポート」

Message Queue 4.0 および 4.1 で導入された機能については、それぞれ、22 ページの「Message Queue 4.0 の新機能」および 18 ページの「Message Queue 4.1 の新機能」を参照してください。

パブリッシャーまたはサブスクライバの複数の送信先

Message Queue 4.2 では、パブリッシャーは複数のトピック送信先にメッセージを発行でき、サブスクライバは複数のトピック送信先からメッセージを消費できます。この機能は、トピック送信先の名前にワイルドカード文字を使用して、複数の送信先を表すことにより実現されます。そのような記号名を使用することにより、管理者は、必要な場合に、ワイルドカードのネーミングスキームに整合する追加のトピック送信先を作成できます。送信先が追加されると自動的に、パブリッシャーはその送信先に発行し、サブスクライバはその送信先から消費するようになります。ワイルドカードトピックのサブスクライバの方が、パブリッシャーよりも一般的です。

注 – この機能は、キュー送信先には適用されません。

トピック送信先の記号名の形式は複数の部分で構成されており、ワイルドカード文字 (*、**、>) で名前の 1 部分または複数の部分を表すことができます。たとえば、次のようなトピック送信先のネーミングスキームがあるとします。

size.color.shape

このトピック名の各部には、次のような値を指定できます。

- *size*: `large`、`medium`、`small` など
- *color*: `red`、 `green`、`blue` など
- *shape*: `circle`、 `triangle`、`square` など

Message Queue では、次のワイルドカード文字がサポートされます。

- *: 単一部分と一致
- **: 1 つ以上の部分と一致
- >: 任意の数の後続部分と一致

従って、複数のトピック送信先を次のように示すことができます。

`large.*.circle` は、次のトピック送信先を表します。

```
large.red.circle
large.green.circle
...
```

**.square は、次のような、.square で終わるすべてのトピック送信先を表します。

```
small.green.square
medium.blue.square
...
```

small.> は、次のような、small. で始まるすべてのトピック送信先を表します。

```
small.blue.circle
small.red.square
...
```

この複数送信先機能を使用するには、前述のようなネーミングスキームを使用してトピック送信先を作成します。クライアントアプリケーションは、それらの送信先記号名を使用してパブリッシャーまたはコンシューマを作成できます。次に例を示します。

```
...
String DEST_LOOKUP_NAME = "large.*.circle";
Topic t = (Destination) ctx.lookup(DEST_LOOKUP_NAME);
TopicPublisher myPublisher = mySession.createPublisher(t)
myPublisher.send(myMessage);

...
String DEST_LOOKUP_NAME = "**.square";
Topic t = (Destination) ctx.lookup(DEST_LOOKUP_NAME);
TopicSubscriber mySubscriber = mySession.createSubscriber(t);
Message m = mySubscriber.receive();
```

最初の例では、ブローカが、記号名 large.*.circle に一致するすべての送信先にメッセージのコピーを配置します。2 番目の例では、記号名 **.square に一致する送信先が 1 つ以上ある場合にサブスクライバが作成され、サブスクライバはその記号名に一致するすべての送信先からメッセージを受信します。記号名に一致する送信先がない場合は、一致する送信先が追加されるまで、サブスクライバは作成されません。

記号名に一致する追加の送信先を管理者が作成すると、その後、その記号名を使用して作成されたワイルドカードパブリッシャーがその送信先にメッセージを発行し、次に、その記号名を使用して作成されたワイルドカードサブスクライバがその送信先からメッセージを受信します。

また、Message Queue 管理ツールでは、トピック送信先のパブリッシャー (プロデューサ) とサブスクライバ (コンシューマ) の合計数のほかに、一致する送信先記号名を含むワイルドカードパブリッシャーの数と、送信先記号名を含むワイルドカードサブスクライバの数も報告されます (存在する場合)。

XML ペイロードメッセージのスキーマ検証

Message Queue 4.2 のこの新機能では、メッセージがブローカに送信された時点で、テキスト (オブジェクトではない) の XML メッセージの XML スキーマを検証できます。XML スキーマ (XSD) の場所は、Message Queue 送信先のプロパティーとして指定されます。XSD の場所が指定されていない場合は、XML ドキュメント内の DTD 宣言を使用して DTD 検証が実行されます。データ型および値の範囲の検証を含む XSD 検証は、DTD 検証よりも厳格です。

クライアントアプリケーションでこの新機能を使用する場合は、Java SE のバージョンを JRE 1.5 以上にアップグレードしてください。

XML スキーマ検証を有効にするには、次の物理送信先プロパティーを設定します。

表 1–5 XML スキーマ検証の物理送信先プロパティー

プロパティー	型	デフォルト値	説明
`validateXMLSchemaEnabled`	ブール型	`false`	XML スキーマ検証が有効になっているかどうか `false` に設定されているか、または何も設定されていない場合、送信先の XML スキーマ検証は有効になっていません。
`XMLSchemaURIList`	文字列	null	XML スキーマドキュメント (XSD) URI 文字列のスペース区切りリスト URI は、XML スキーマ検証が有効になっている場合に使用する 1 つ以上の XSD の場所を示します。 複数の URI を指定する場合は、この値を二重引用符で囲みます。 例: “`http://foo/flap.xsd http://test.com/test.xsd`” このプロパティーが設定されていないか、または null の場合に XML 検証が有効になっていると、XML ドキュメント内で指定された DTD を使用して XML 検証が実行されます。
`reloadXMLSchemaOnFailure`	ブール型	`false`	障害発生時に XML スキーマを再読み込みするかどうか false に設定されているか、または何も設定されていない場合は、検証で障害が発生したときにスキーマの再読み込みは行われません。

XML 検証が有効になっている場合、Message Queue クライアントランタイムは、XML メッセージをブローカに送信する前に、指定された XSD (XSD が指定されていない場合は DTD) に照らしたメッセージの検証を試みます。指定されたスキーマが見つからないか、またはメッセージを検証できない場合は、メッセージは送信されずに、例外がスローされます。

送信先の作成時または更新時に、`imqcmd create dst` または `imqcmd update dst` コマンドを使用して、XML 検証プロパティーを設定できます。XML 検証プロパティーは、送信先がアクティブでないときに設定してください。つまり、送信先のコンシューマとプロデューサがないとき、および送信先にメッセージが存在しないときです。

注 – 実行時に XSD にアクセスできない場合は、送信先がアクティブなときに `XMLSchemaURIList` を変更しなければならないことがあります。

たとえばプロデューサが送信先に接続されている場合など、送信先がアクティブなときに XML 検証プロパティーのいずれかが設定されると、その変更は、プロデューサがブローカに再接続されるまで有効になりません。同様に、アプリケーションの要件 が変更されたために XSD が変更された場合は、変更後の XSD に基づいて XML メッセージを生成するすべてのクライアントアプリケーションをブローカに再接続してください。

`reloadXMLSchemaOnFailure` プロパティーが `true` に設定されているときに XML 検証に失敗すると、Message Queue クライアントランタイムは、メッセージを再度検証する前に XSD の再読み込みを試みます。再読み込みした XSD を使用した検証に失敗すると、クライアントランタイムは例外をスローします。

分散トランザクションの C-API サポート

X/Open 分散トランザクションモデルに従って、分散トランザクションのサポートは、1 つ以上のリソースマネージャーで実行される操作の追跡と管理を行う分散トランザクションマネージャーに依存します。Message Queue 4.2 では、Message Queue C-API で XA インタフェース (分散トランザクションマネージャーと、XA 準拠のリソースマネージャーとしての Message Queue の間のインタフェース) がサポートされるようになりました。それにより、BEA Tuxedo などの分散トランザクション処理環境で実行される Message Queue C-API クライアントが、分散トランザクションに参加できるようになりました。

この分散トランザクションのサポートは、次に示す、XA インタフェース仕様を実装するための新しい C-API 関数 (および新しいパラメータとエラーコード) から成ります。

```
MQGetXAConnection()
MQCreateXASession()
```

C クライアントアプリケーションを分散トランザクションのコンテキストで使用する場合は、MQGetXAConnection() を使用して接続を取得し、MQCreateXASession() を使用して、メッセージを生成および消費するためのセッションを作成します。 すべての分散トランザクションの開始、コミット、およびロールバックは、分散トランザクションマネージャーが提供する API によって管理されます。

public 情報

X/Open XA インタフェース仕様では、Message Queue の XA 準拠リソースマネージャーに関する次の public 情報が必要です。

- `xa_switch_t` 構造体の名前: `sun_my_xa_switch`
- リソースマネージャーの名前: `SUN_RM`
- リンクする MQ C-API ライブラリ: `mqcrt`
- `xa_close` 文字列と形式: なし
- `xa_open` 文字列と形式: 「;」で区切られた名前と値のペア (「名前=値」形式)

次の名前と値の組み合わせがサポートされます。

表 1–6　Message Queue リソースマネージャーの名前と値の組み合わせ

名前	値	説明	デフォルト
address	*host*:*port*	ブローカの Portmapper サービスのホストとポート	`localhost:7676`
username	文字列	ブローカへの接続に使用するユーザー名	`guest`
password	文字列	ユーザー名のパスワード	`guest`
conntype	TCP または SSL	ブローカへの接続のプロトコルの種類	`TCP`
trustedhost	true/false	ブローカのホストが信頼できるかどうか (conntype が SSL の場合にのみ使用)	`true`
certdbpath	文字列	NSS 証明書およびキーデータベースファイルが格納されたディレクトリへのフルパス	なし
clientid	文字列	JMS 永続サブスクリプションでのみ必要	なし

表 1–6 Message Queue リソースマネージャーの名前と値の組み合わせ (続き)

名前	値	説明	デフォルト
reconnects	整数	ブローカへの再接続の試行回数 (0 は再接続しないことを意味する)	0

プログラミング例

分散トランザクションを使用するアプリケーションのプログラミングでは、トランザクションマネージャー環境で動作するサーバー側のサービスと、トランザクションマネージャー API を呼び出すクライアント側のコードを作成します。Message Queue 4.2 には、Tuxedo トランザクションマネージャーに基づくプログラミング例が用意されています。それらの例は、各プラットフォームの `./C/tuxedo` ディレクトリ内の sample programs ディレクトリにあります。

このディレクトリに含まれている `README` ファイルに、Message Queue リソースマネージャーを使用するための Tuxedo のセットアップ方法、および Tuxedo 環境で次のプログラミング例を構築する方法が記載されています。

プログラミング例	説明
`jmsserver.c`	Message Queue を使用してメッセージを送受信する Tuxedo サービスを実装します。
`jmsclient_sender.c`	`jmsserver.c` プログラムのメッセージ生成サービスを使用する Tuxedo クライアントです。
`jmsclient_receiver.c`	`jmsserver.c` プログラムのメッセージ受信サービスを使用する Tuxedo クライアントです。
`async_jmsserver.c`	Message Queue を使用してメッセージを非同期に消費する Tuxedo サービスを実装します。
`jmsclient_async_receiver.c`	`async_jmsserver.c` プログラムの非同期メッセージ消費サービスを使用する Tuxed クライアントです。

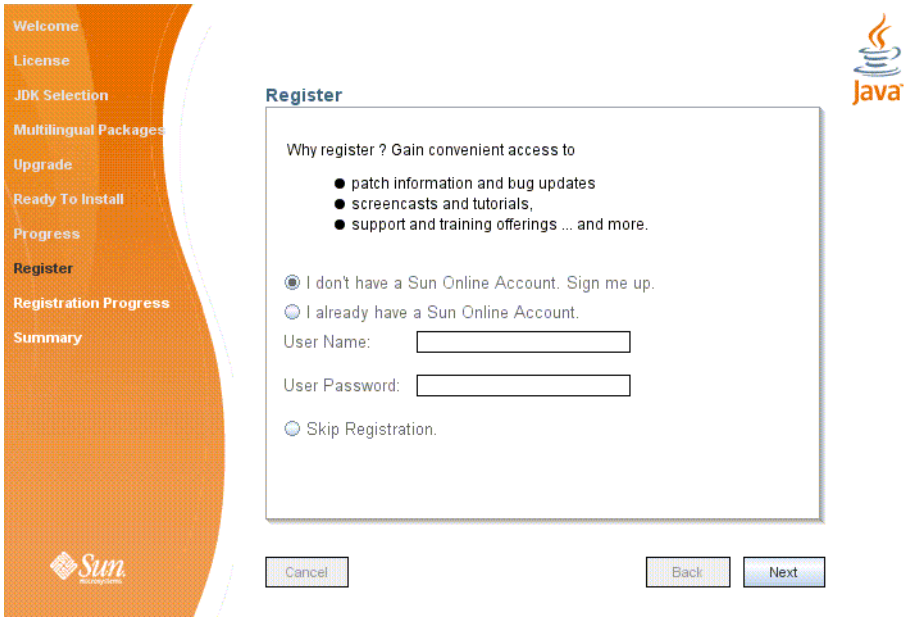
インストーラでの **Sun Connection** 登録のサポート

Message Queue インストーラが、Message Queue を Sun Connection に登録できるように拡張されました。Sun Connection は、Sun のハードウェアとソフトウェアの追跡、構成、および維持を支援するために Sun が提供するサービスです。

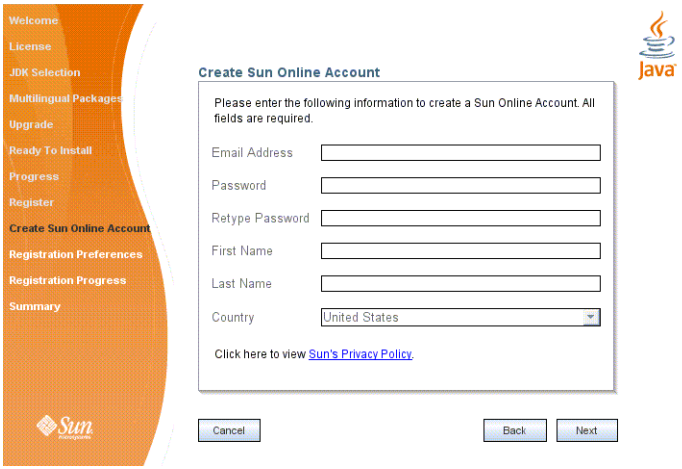
Message Queue のインストール時に、Message Queue を Sun Connection に登録できます。インストールした Message Queue に関する情報、たとえば、リリースバージョン、ホスト名、オペレーティングシステム、インストール日などの基本情報は、Sun Connection のデータベースに安全に転送されます。Sun Connection のインベントリサービスは、Sun のハードウェアとソフトウェアの構成 に役立ちます。ま

た、更新サービスでは、最新のセキュリティー修正、推奨される更新、および機能拡張に関する情報を得ることができます。

Message Queue 4.2 では、Sun Connection 登録用に、次のインストーラ画面が追加されています。



登録するには、Sun Online アカウントを持っているか、または作成する必要があります。アカウントをまだ持っていない場合は、Sun Online アカウントを作成するための、次の画面が表示されます。



インストール時にこれらの画面を使用して Message Queue を登録できますが、インストールが完了した後で登録することもできます。その場合は、次のように登録専用モードでインストーラを実行します。

```
# installer -r
```

登録専用モードを使用するには、Message Queue 4.2 がすでにインストールされている必要があります。登録専用モードでは、登録に関連するインストーラ画面のみが表示されます。

MySQL データベースのサポート

Message Queue 4.2 では、JDBC ベースのデータストアとして MySQL データベースがサポートされます。スタンドアロンブローカ用の JDBC データベースとしては、MySQL Cluster Edition を使用できます。MySQL Cluster Edition は、高可用性ブローカクラスタに必要な高可用性共有データストアとして使用することもできます。MySQL を使用するための Message Queue の構成については、『Sun Java System Message Queue 4.2 Administration Guide』の「Configuring a JDBC-Based Data Store」および『Sun Java System Message Queue 4.2 Administration Guide』の「High-Availability Cluster Properties」を参照してください。

Message Queue 4.1 の新機能

Message Queue 4.1 は、いくつかの新機能、機能拡張、およびバグ修正を実装したマイナーリリースです。ここでは 4.1 リリースの新機能について説明するとともに、詳細な情報の参照先を示します。

- 19 ページの「高可用性ブローカクラスタ」
- 20 ページの「JAAS サポート」

- 20 ページの「持続データストアの形式の変更」
- 20 ページの「ブローカの環境設定」
- 21 ページの「Java ES Monitoring Framework のサポート」
- 21 ページの「拡張されたトランザクション管理」
- 21 ページの「C クライアント接続の固定ポート」

Message Queue 4.0 で導入された機能については、22 ページの「Message Queue 4.0 の新機能」を参照してください。

高可用性ブローカクラスタ

Message Queue 4.1 では、高可用性ブローカクラスタが導入されました。従来のブローカクラスタでは、ブローカで障害が発生した場合に別のブローカがメッセージングサービスを提供する「メッセージングサービス」の可用性のみが提供されていました。高可用性ブローカクラスタでは、ブローカで障害が発生した場合に持続メッセージと状態データを使用して別のブローカがメッセージ配信を継承できる、「データ」の可用性も提供されます。

Message Queue 4.1 で導入された高可用性の実装では、JDBC ベースの共有データストアが使用されます。ブローカクラスタ内の各ブローカがそれぞれの持続データストアを持つのではなく、クラスタ内のすべてのブローカが同じ JDBC 準拠データベースを共有します。特定のブローカで障害が発生すると、そのブローカのメッセージルーティングおよび配信を、メッセージクラスタ内の別のブローカが継承します。そのときに、継承ブローカは、共有データストアのデータおよび状態情報を使用します。障害が発生したブローカのメッセージングクライアントは継承ブローカに再接続するため、メッセージングサービスは中断されません。

Message Queue 4.1 の高可用性実装に使用される JDBC ベースの共有ストアは、それ自体も高可用性ストアでなければなりません。高可用性データベースを持っていない場合、またはメッセージ配信が中断されないことを重要視しない場合は、データの可用性がなくサービスの可用性のみを提供する従来のクラスタを引き続き使用できます。

Message Queue 4.1 の高可用性ブローカクラスタを設定するには、クラスタ内のブローカごとに、次のブローカプロパティーを指定します。

- クラスタメンバーシッププロパティー。ブローカが高可用性ブローカクラスタ内にあることと、クラスタの ID およびクラスタ内のブローカの ID を指定します。
- 高可用性データベースプロパティー。持続データモデル (JDBC)、データベースベンダーの名前、およびベンダー固有の設定プロパティーを指定します。
- 障害検出および継承プロパティー。ブローカ障害の検出および継承ブローカによる処理の方法を指定します。

高可用性ブローカクラスタ実装を使用するには、次の操作を実行します。

1. 高可用性データベースをインストールします。

2. JDBC ドライバの .jar ファイルをインストールします。
3. 高可用性持続データストア用のデータベーススキーマを作成します。
4. クラスタ内のブローカごとに、高可用性プロパティーを設定します。
5. クラスタ内の各ブローカを起動します。

高可用性ブローカクラスタの概念に関する説明、および従来のクラスタとの比較については、『Sun Java System Message Queue 4.2 Technical Overview』の第 4 章「Broker Clusters」を参照してください。高可用性ブローカクラスタの手続きおよび参照に関する情報については、『Sun Java System Message Queue 4.2 Administration Guide』の第 10 章「Configuring and Managing Broker Clusters」および『Sun Java System Message Queue 4.2 Administration Guide』の「Cluster Configuration Properties」を参照してください。

Message Queue 4.0 で高可用性データベースを使用していた場合、高可用性ブローカクラスタに切り替えるには、データベースマネージャーユーティリティー (`imqdbmgr`) を使用して共有持続データストアに変換できます。49 ページの「ブローカクラスタ」で、既知の問題および制限事項についても参照してください。

JAAS サポート

Message Queue 4.1 では、ファイルベースおよび LDAP ベースの組み込み認証機構に加えて、JAAS (Java Authentication and Authorization Service) サポートも導入されています。JAAS を使用すると、外部の認証機構をブローカに接続して Message Queue クライアントを認証できます。

ブローカによって JAAS 準拠の認証サービスで利用可能になる情報についての説明、およびそれらのサービスを使用するようにブローカを設定する方法については、『Sun Java System Message Queue 4.2 Administration Guide』の「Using JAAS-Based Authentication」を参照してください。

持続データストアの形式の変更

Message Queue 4.1 は、JDBC ベースのデータストアが高可用性ブローカクラスタをサポートするように変更されました。このため、JDBC ベースのデータストアの形式が Version 410 になりました。Version 350、370、および 400 の形式は、自動的に Version 410 に移行されます。

ファイルベースの持続データストアの形式は、何も変更されていないので、Version 370 のままです。

ブローカの環境設定

Message Queue 4.1 の環境設定ファイル `imqenv.conf` に、プロパティー `IMQ_DEFAULT_EXT_JARS` が追加されました。このプロパティーを設定して、ブローカが起動するときに `CLASSPATH` に含まれる外部 .jar ファイルのパス名を指定できます。このプロパティーを使用して外部 .jar ファイルの場所を指定した場合は、これらの

ファイルを lib/ext ディレクトリにコピーする必要はなくなります。外部 .jar ファイルは、JDBC ドライバまたは JAAS ログインモジュールを参照できます。次のサンプルプロパティーは、JDBC ドライバの場所を指定します。

```
IMQ_DEFAULT_EXT_JARS=/opt/SUNWhadb4/lib/hadbjdbc4.jar:/opt/SUNWjavadb/derby.jar
```

Java ES Monitoring Framework のサポート

Message Queue 4.1 では、 Sun Java Enterprise System (Java ES) Monitoring Framework のサポートが導入されました。Monitoring Framework は、一般的なグラフィカルインタフェースを使用して Java ES コンポーネントを監視できるようにします。このインタフェースは、 Sun Java System Monitoring Console と呼ばれる Web ベースのコンソールによって実装されます。管理者はこのコンソールを使用して、パフォーマンス統計の表示、自動監視のためのルールの作成、アラームの確認などを行えます。ほかの Java ES コンポーネントと一緒に Message Queue を実行している場合は、1 つのインタフェースを使用してすべてのコンポーネントを管理するほうが便利なことがあります。

Java ES Monitoring Framework による Message Queue の監視については、XREF を参照してください。

拡張されたトランザクション管理

以前は、管理上、PREPARED 状態のトランザクションだけをロールバックすることができました。つまり、分散トランザクションの一部であるセッションが正常に終了しなかった場合、そのトランザクションは、管理者がクリーンアップできない状態のままになりました。Message Queue 4.1 では、コマンドユーティリティー (imqcmd) を使用して、STARTED、FAILED、INCOMPLETE、COMPLETE、PREPARED の状態にあるトランザクションをクリーンアップ (ロールバック) できます。

特定のトランザクションがロールバックできるかどうか (特に、PREPARED 状態でない場合) の判別に役立つように、このコマンドユーティリティーは、追加のデータを imqcmd query txn 出力の一部として提供します。このデータは、そのトランザクションを開始した接続のコネクション ID を提供し、トランザクションが作成された時間を示します。管理者は、この情報を使用して、トランザクションをロールバックする必要があるかどうかを決定できます。一般に、管理者は早計にトランザクションをロールバックすることを避けるとよいでしょう。

C クライアント接続の固定ポート

Message Queue 4.1 では、C クライアントは、Java クライアントと同様に、ブローカのポートマッパーサービスで動的に割り当てられるポートではなく、固定ブローカポートに接続できるようになりました。固定ポート接続は、ファイアウォールを経由する場合、または他の何らかの理由でポートマッパーサービスをバイパスする必要がある場合に役立ちます。

固定ポート接続を設定するには、ブローカと C クライアントランタイムの両方 (接続の両端) を設定する必要があります。たとえば、ssljms を介してクライアントをポート 1756 に接続する場合は、次のように操作します。

- クライアント側で、次のプロパティーを設定します。

  ```
  MQ_SERVICE_PORT_PROPERTY=1756
  MQ_CONNECTION_TYPE_PROPERTY=SSL
  ```

- ブローカ側で、imq.*serviceName.protocolType*.port プロパティーを次のように設定します。

  ```
  imq.ssljms.tls.port=1756
  ```

注 – MQ_SERVICE_PORT_PROPERTY 接続プロパティーは、Message Queue 3.7 Update 2 にバックポートされています。

Message Queue 4.0 の新機能

Message Queue 4.0 は、Application Server 9 PE のサポートに限定されたマイナーリリースです。このリリースでは、いくつかの新機能、機能拡張、およびバグ修正が実装されています。この節では、このリリースの新機能について説明します。

注意 – version 4.0 には、軽度とはいえ、状況によって十分な対応が必要になる変更が導入されました。その 1 つとして、パスワードを指定するコマンド行オプションが使用されなくなったことが挙げられます。今後は、46 ページの「使用されなくなったパスワードオプション」で説明するように、すべてのパスワードをファイルに保存するか、または要求されたときに入力します。

JMX 管理 API のサポート

Message Queue 4.0 には、Java Management Extensions (JMX) 仕様に準拠して、Message Queue ブローカを設定および監視するための新しい API が追加されました。この API を使用すると、Java アプリケーション内部からプログラムによってブローカ関数を

設定および監視することができます。以前のバージョンの Message Queue では、これらの関数にはコマンド行管理ユーティリティーまたは管理コンソールからしかアクセスできませんでした。

詳細については、『Sun Java System Message Queue 4.2 Developer's Guide for JMX Clients』を参照してください。

クライアントランタイムログ

Message Queue 4.0 では、接続およびセッション関連のイベントに関するクライアントランタイムログのサポートが導入されました。

クライアントランタイムログおよびその設定方法については、『Java Dev Guide』の 137 ページを参照してください。

接続イベント通知 API

Message Queue 4.0 にはイベント通知 API が導入され、クライアントランタイムが接続状態の変更をアプリケーションに通知できるようになりました。接続イベント通知を使用すると、Message Queue クライアントは接続のクローズおよび再接続イベントを待機して、通知タイプと接続状態に基づいて適切なアクションを起こすことができます。たとえば、フェイルオーバーが発生してクライアントが別のブローカに再接続された場合、アプリケーションはそのトランザクションの状態をクリーンアップしてから新しいトランザクションに進む必要があるかもしれません。

接続イベント、およびイベントリスナの作成方法については、『Java Dev Guide』の 96 ページを参照してください。

ブローカ管理の拡張機能

Message Queue 4.0 では、コマンドユーティリティー (`imqcmd`) に、サブコマンドといくつかのコマンドオプションが追加されました。管理者はこれらを使用して、ブローカを休止したり、指定した間隔の後でブローカをシャットダウンしたり、接続を破棄したり、Java システムプロパティー (たとえば、コネクション関連のプロパティー) を設定したりできます。

- ブローカを休止すると休止状態になり、ブローカをシャットダウンまたは再起動する前に、メッセージを排出してしまうことができます。休止状態にあるブローカに新しく接続が作成されることはありません。ブローカを休止するには、次のようなコマンドを入力します。

  ```
  imqcmd quiesce bkr -b Wolfgang:1756
  ```

- 指定した間隔の後でブローカをシャットダウンするには、次のようなコマンドを入力します。(時間間隔は、ブローカをシャットダウンするまでの待機を秒数で指定します。)

  ```
  imqcmd shutdown bkr -b Hastings:1066 -time 90
  ```

時間間隔を指定した場合、ブローカはシャットダウンが発生するタイミングを示すメッセージを記録します。次にその例を示します。

```
Shutting down the broker in 29 seconds (29996 milliseconds)
```

ブローカがシャットダウンを待っている間、ブローカの動作は次のような影響を受けます。

- 管理 JMS 接続は引き続き受け付けられます。
- 新しい JMS 接続は受け付けられません。
- 既存の JMS 接続は引き続き機能します。
- ブローカが高可用性ブローカクラスタ内のほかのブローカを継承することはできません。
- imqcmd ユーティリティーはブロックはせず、シャットダウンの要求をブローカに送信してすぐに返します。

- 接続を破棄するには、次のようなコマンドを入力します。

```
imqcmd destroy cxn -n 2691475382197166336
```

コマンド `imqcmd list cxn` または `imqcmd query cxn` を使用してコネクション ID を取得します。

- `imqcmd` を使用してシステムプロパティーを設定するには、新しい –D オプションを使用します。これは、JMS コネクションファクトリのプロパティー、またはコネクション関連の Java システムのプロパティーの設定または上書きに便利です。次に例を示します。

```
imqcmd list svc -secure -DimqSSLIsHostTrusted=true
imqcmd list svc -secure -Djavax.net.ssl.trustStore=/tmp/mytruststore
            -Djavax.net.ssl.trustStorePassword=mytrustword
```

`imqcmd` コマンドの構文の詳細は、『Sun Java System Message Queue 4.2 Administration Guide』の第 15 章「Command Line Reference」を参照してください。

JDBC ベースのデータストアに関する情報の表示

Message Queue 4.0 では、データベースマネージャーユーティリティー `imqdbmgr` に、新しい `query` サブコマンドが追加されました。このサブコマンドは、JDBC ベースのデータストアに関する情報 (データベースバージョン、データベースユーザー、データベーステーブルが作成されたかどうかなど) を表示するために使用します。

次に、このコマンドによって表示される情報の例を示します。

```
imqdbmgr query

[04/Oct/2005:15:30:20 PDT] Using plugged-in persistent store:
        version=400
        brokerid=Mozart1756
```

```
        database connection url=jdbc:oracle:thin:@Xhome:1521:mqdb
        database user=scott
Running in standalone mode.
Database tables have already been created.
```

JDBC プロバイダのサポート

Message Queue 4.0 では、Apache Derby Version 10.1.1 が、JDBC ベースのデータストアプロバイダとしてサポートされるようになりました。

持続データストアの形式の変更

Message Queue 4.0 では、最適化のため、および今後の拡張機能をサポートするために、JDBC ベースのデータストアに変更が加えられました。このため、JDBC ベースのデータストアの形式が Version 400 になりました。Message Queue 4.0 では、ファイルベースのデータストアのバージョンは、何も変更されていないので、370 のままです。

追加のメッセージプロパティー

Message Queue 4.0 では、デッドメッセージキューに配置されたすべてのメッセージで設定される 2 つの新しいプロパティーが追加されました。

- `JMS_SUN_DMQ_PRODUCING_BROKER` は、メッセージが生成されたブローカを指定します。
- `JMS_SUN_DMQ_DEAD_BROKER` は、メッセージをデッドとして指定したブローカを指定します。

SSL サポート

Message Queue 4.0 から、クライアントコネクションファクトリのプロパティー `imqSSLIsHostTrusted` のデフォルト値が `false` になりました。使用しているアプリケーションが以前のデフォルト値の `true` に依存している場合は、再設定を行い、プロパティーを明示的に `true` に設定する必要があります。

ブローカが自己署名付き証明書を使用するように設定されているときは、ホストを信頼することもできます。この場合、接続で SSL ベースの接続サービスを使用するように指定する (imqConnectionType プロパティーを使用する) ことに加えて、`imqSSLIsHostTrusted` プロパティーを true に設定することをお勧めします。

たとえば、ブローカが自己署名付き証明書を使用するときに安全にクライアントアプリケーションを実行するには、次のようなコマンドを使用します。

```
java -DimqConnectionType=TLS
      -DimqSSLIsHostTrusted=true ClientAppName
```

ブローカが自己署名付き証明書を使用するときにコマンドユーティリティー (imqcmd) を安全に使用するには、次のようなコマンドを使用します (コネクタサービスのリストを表示する場合)。

```
imqcmd list svc -secure -DimqSSLIsHostTrusted=true
```

今後のリリースで使用されなくなる機能

メッセージベースの監視では、メトリックトピック送信先に書き込まれたメトリック (計測データ) 情報を使用してブローカとその送信先を管理できますが、この機能は今後のリリースでは使用されなくなります。

メッセージベースの監視では、ブローカの設定可能なメトリックスメッセージプロデューサを利用して、メトリックデータを JMS メッセージに書き込みます。この JMS メッセージは、メッセージに含まれているメトリック情報の種類に応じてメトリックトピック送信先に送信されます。このメトリック情報にアクセスするには、適切なメトリックトピック送信先へのサブスクライブ、そのメッセージの消費、および必要に応じたデータ処理を行うクライアントアプリケーションを記述します。

メッセージベースの監視機能の代わりに、MQ 4.0 で実装された JMX 管理 API が使用されるようになっています (22 ページの「JMX 管理 API のサポート」を参照)。この JMX API は、より包括的 (トピック送信先に書き込まれるよりも多くのメトリックデータが含まれる) であり、JMX 業界標準に基づいています。

Message Queue で JMX API がサポートされている現在、あえてメッセージベースの監視を使用する理由はありません。メッセージベースの監視機能に関する情報は、この機能が正式に使用されなくなるまで Message Queue ドキュメントに残されます。

Message Queue 4.2 で修正されたバグおよび最近のリリース

Message Queue 4.2 には、新しいバグ修正が施され、Message Queue 4.1 および Message Queue 4.0 の各リリースでのバグ修正も組み込まれています。

次の各節では、それぞれのリリースで修正されたバグの一覧を示します。

Message Queue 4.2 で修正されたバグ

次の表に、Message Queue 4.2 で修正されたバグを示します。

表 1–7　Message Queue 4.2 で修正されたバグ

バグ	説明
6581592	インストーラまたはアンインストーラがテキストモード (`installer –t`) で実行されているときは、「概要」画面にログファイルまたは概要ファイルを含むディレクトリが表示されますが、これらのファイルの名前のリストは表示されません。
6585911	インストーラにバンドルされている、インストーラの実行に使用する JRE が、インストーラの「JDK の選択」画面に正しく表示されません。
6587112	複数バイトロケールで、インストーラの「概要」画面に無意味な文字が表示されます。
6587127	回答ファイルを参照して (`installer -a` *filename* `-s`) インストーラを実行するときに、回答ファイルが存在しない場合、矛盾した不明瞭なエラーメッセージが表示されます。
6590969	クライアント接続認証で、DN ユーザー名形式が許可されます。
6594381	Message Queue 4.1 ローカリゼーション RPM のインストール (「多言語パッケージ」画面で「Message Queue 多言語パッケージのインストール」チェックボックスにチェックマークを付けた場合に実行される) は、古いバージョンの Message Queue ローカリゼーション RPM がシステムに存在すると、異常終了します。
6599144	Message Queue 4.2 のアンインストール時に、Java SE 6 ではスプラッシュ画面とアンインストーラがハングアップして空のグレー画面になりますが、Java SE 5 では正しく動作します。
6615741	ロールバックの前に元のコンシューマが閉じられると、ロールバックされたトランザクションコンシューマセッションで配信されたメッセージは再配信されません。
6629922	分散トランザクションハンドラは、非アクティブなコンシューマ に正しい順序でメッセージを再配信しません。
6635130	送信先のメモリー制限またはメッセージ制限に達したために一時停止された後、ブローカは、非持続メッセージのプロデューサに 生成を再開するように通知できません。
6641117	ロールバックの後に元のコンシューマが閉じられると、ロールバックされたトランザクションコンシューマセッションで配信されたメッセージは再配信されません。
6683897	設定が正常に完了したように見える場合でも、Message Queue インストーラの「概要」画面で設定エラーが報告されます。インストーラは、いくつかのコンピュータの `/dev/sterr` に書き込むことができません。
6684069	コンシューマトランザクションで多数のメッセージがリモートクライアントに配信されるブローカクラスタで、トランザクションのコミットに障害が発生します。
6688935	Portmapper の読み取りのタイムアウトが短すぎます。

表 1–7 Message Queue 4.2 で修正されたバグ (続き)

バグ	説明
6695238	C クライアントアプリケーションは、パスにスペースが含まれている場所にインストールされたブローカに接続できません。
6710168	送信先が再開されないまま 2 回続けて一時停止されると、コンシューマはメッセージを消費しなくなります。
6710169	JMX 操作 `ConsumerManagerMonitor.getConsumerInfo` は、通知モードでは常に `SESSION_TRANSACTED` を返します。

Message Queue 4.1 で修正されたバグ

次の表に、Message Queue 4.1 で修正されたバグを示します。

表 1–8 Message Queue 4.1 で修正されたバグ

バグ	説明
6381703	処理されたリモートメッセージが、そのメッセージの発信元のブローカが再起動された場合に、2 回コミットされることがあります。
6388049	未完了の分散トランザクションをクリーンアップできません。
6401169	imqcmd のコミットおよびロールバックオプションで、確認のプロンプトが表示されません。
6473052	自動作成されたキューのデフォルトはラウンドロビンであるべきです。(`MaxNumberConsumers` = -1)。
6474990	ブローカのログに、`imqcmd list dst` コマンドに対して `ConcurrentModificationException` が示されます。
6487413	LimitBehavior が `REMOVE_OLDEST` または `REMOVE_LOWER_PRIORITY` のときに、メモリーがリークします。
6488340	ブローカがスピンし、クライアントは確認に対する応答を待機します。
6502744	ブローカが、デッドメッセージキューのデフォルト制限 (1000 のメッセージ) を尊重しません。
6517341	クライアントが高可用性ブローカクラスタへ接続している際の、再接続ロジックを、*imqReconnectEnabled* プロパティーの値に関係なく再接続を可能にするように改善すべきです。
6528736	起動時に Windows 自動起動サービス (*imqbrokersvc*) に障害が発生します。
6561494	両者が 1 つのセッションを共有していると、メッセージが間違ったコンシューマに配信されます。

表 1–8 Message Queue 4.1 で修正されたバグ (続き)

バグ	説明
6567439	PREPARED トランザクションの生成メッセージがブローカの再起動後にコミットされた場合、それらのメッセージは順序どおりに配信されません。

Message Queue 4.0 で修正されたバグ

次の表に、Message Queue 4.0 で修正されたバグを示します。

表 1–9 Message Queue 4.0 で修正されたバグ

バグ番号	説明
4986481	Message Queue 3.5 では、自動再接続モードでの Session.recover の呼び出しがハングすることがありました。
4987325	Session.recover の呼び出し後に、再配信されたフラグは再配信されたメッセージに対して false に設定されました。
6157073	新しく接続メッセージが変更され、接続の合計数のほかにサービス上の接続数も表示されるようになりました。
6193884	Message Queue は、メッセージに非アスキー文字を使用するロケールで文字化けメッセージを syslog に出力します。
6196233	JMSMessageID を使用したメッセージ選択が機能しません。
6251450	クラスタのシャットダウン時、 connectList に ConcurrentModificationException が発生します。
6252763	java.nio.HeapByteBuffer.putLong/Int に java.nio.BufferOverflowException が発生します。
6260076	Oracle ストレージを使用すると、起動後に発行される最初のメッセージが遅くなります。
6260814	JMSXUserID 上のセレクタ処理が、常に false と評価されてしまいます。
6264003	キューブラウザにコミットされていないメッセージが表示されます。
6271876	消費されていないメッセージを含むコンシューマを閉じようとすると、接続フロー制御が正しく動作しなくなります。
6279833	Message Queue では、2 つのブローカが同じ JDBC テーブルを使用することはできません。
6293053	システム IP アドレスが変更された場合、–reset store を使用してストアをクリアしていないと、マスターブローカが正しく起動しません。
6294767	Message Queue ブローカがネットワークソケットを開く場合、そのネットワークソケット上に SO_REUSEADDR を設定する必要があります。

表 1–9　Message Queue 4.0 で修正されたバグ　　(続き)

バグ番号	説明
6304949	TopicConnectionFactory に ClientID プロパティーを設定することができません。
6307056	txn ログがパフォーマンス上のボトルネックになっています。
6320138	Message Queue C API では reply-to ヘッダーからキューの名前を決定することができません。
6320325	Solaris 上に JDK 1.4 と JDK 1.5 の両方がインストールされている場合、ブローカが JDK 1.5 より前に JDK 1.4 を検出してしまうことがあります。
6321117	マルチブローカクラスタの初期化で java.lang.NullPointerException が発生します。
6330053	サブスクライバからトランザクションをコミットしている場合、JMS クライアントで java.lang.NoClassDefFoundError が発生します。
6340250	C-API で MESSAGE タイプをサポート。
6351293	Apache Derby データベースへのサポートを追加。

Message Queue 4.2 のマニュアルの更新

この節では、Message Queue 4.2 のマニュアルの更新について説明します。

互換性の問題

この節では、Message Queue 4.2 に関する互換性の問題を説明しています。

インタフェースの安定性

Sun Java System Message Queue で使用される多くのインタフェースは、時間の経過につれて変更される可能性があります。『Sun Java System Message Queue 4.2 Administration Guide』の付録 B「Stability of Message Queue Interfaces」では、インタフェースをそれらの安定性に従って分類しています。安定性に優れるインタフェースほど、製品の後継バージョンで変更される可能性は低くなります。

Message Queue の次回のメジャーリリースに関係する問題

Message Queue の次回のメジャーリリースでは、現在の Message Queue クライアントアプリケーションがそのリリースと互換性がなくなるような変更が導入される可能性がありますこの情報は、完全な情報開示の目的で提供しています。

- Sun Java System Message Queue の一部としてインストールされる各ファイルの場所が変更される可能性があります。これによって、特定の Message Queue ファイルの現在の場所に依存する既存のアプリケーションの動作が中断する可能性があります。
- Message Queue 3.5 以前のブローカは、これより新しいブローカのクラスタ内では動作できなくなる可能性があります。
- 今後のリリースでは、Message Queue クライアントは 1.5 より前のバージョンの JDK を使用できなくなる可能性があります。
- 今後のリリースでは、Message Queue クライアントは 1.6 より前のバージョンの JDK を使用できなくなる可能性があります。

Message Queue 4.2 ドキュメントセットの変更点

Message Queue 4.2 ドキュメントセットは、次に説明するように、Message Queue 4.1 ドキュメントセットから更新されています。

『技術の概要』

『Sun Java System Message Queue 4.2 Installation Guide』 は、Message Queue 4.2 の新機能および高可用性ブローカクラスタの最新フレームワークを反映するように更新されました。

『管理ガイド』

『管理ガイド』は、Message Queue 4.2 の新機能を反映するように更新されました。

インストールとアップグレードに関する情報

『Sun Java System Message Queue 4.2 Installation Guide』 は、Message Queue 4.2 の新機能、特に、インストーラの新しい Sun Connection 登録機能を反映するように更新されました。この情報は、『Message Queue リリースノート』でも提供しています。

『Developer's Guide for Java Clients』

Developer's Guide for Java Clients は、Message Queue 4.2 の新機能を反映するように更新されました。この情報は、『Message Queue リリースノート』でも提供しています。

『Developer's Guide for C Clients』

Developer's Guide for C Clients は、 Message Queue 4.2 の新機能を反映するように更新されました。この情報は、『Message Queue リリースノート』でも提供しています。

『Developer's Guide for JMX Clients』

Message Queue 4.2 の新機能を反映する更新は行われていません。この情報は、『Message Queue リリースノート』でも提供しています。

新しい送信先メトリック

Message Queue 4.2 には、ブローカクラスタでの送信先の監視に役立つ新しい送信先メトリックが含まれています。ブローカクラスタでは、送信先はクラスタ内のすべてのブローカに伝達されます。ただし、メッセージは、生成されたときに、メッセージプロデューサのホームブローカのターゲット送信先に格納され、その送信先のアクティブコンシューマが存在する場合のみ、クラスタ内の別のブローカの対応する送信先に送信されます。このため、指定された送信先に格納されたメッセージは、送信先が属するクラスタ内のブローカに依存します。

つまり、ブローカクラスタで、クラスタ内の特定のブローカ上の特定の送信先に格納されるメッセージには、直接その送信先に対して生成されたメッセージと、クラスタ内のリモートブローカからその送信先に送信されるメッセージがあります。ブローカクラスタでのメッセージのルーティングと配信を分析する際に、送信先のメッセージのうち、ローカルで生成されたメッセージの数と、リモートで生成されたメッセージの数を調べると訳に立つことがあります。

次の表に、Message Queue 4.2 に含まれる 2 つの新しい物理送信先メトリック量を示します。新しいメトリック量は、`imqcmd list dst` と `imqcmd query dst` コマンド、および新しい JMX 属性 (36 ページの「送信先監視 MBean」 を参照) を介して利用できます。

表 1–10 物理送信先のメトリック

メトリック量	説明	ログファイル	metrics dst メトリックタイプ	メトリックトピック
Num messages remote	メモリーおよび持続ストアに格納されている、クラスタ内のリモートブローカに対して生成されたメッセージの現在の数。この数には、トランザクションに含まれているメッセージは含まれません。	不可	使用不可[1]	使用不可
Total message bytes remote	メモリーおよび持続ストアに格納されている、クラスタ内のリモートブローカに対して生成されたメッセージの合計サイズ (バイト単位)。この値には、トランザクションに含まれているメッセージは含まれません。	不可	使用不可[1]	使用不可

[1] `imqcmd query dst` コマンドで使用可能

Solaris 10 OS でのブローカの自動起動

この節では、Solaris 10 オペレーティングシステムでブローカの自動起動を設定する方法について説明します。コンピュータの再起動時に rc ファイルを使用してブローカの自動起動を実装するのではなく、次の手順では Solaris 10 サービス管理機能 (SMF) を使用します。

サービス管理機能の使用の詳細については、Solaris 10 のドキュメントを参照してください。

▼ Solaris 10 OS でブローカの自動起動を実装する

1 **mqbroker サービスを SMF リポジトリにインポートします。**

```
# svccfg import /var/svc/manifest/application/sun/mq/mqbroker.xml
```

2 **mqbroker サービスの状態をチェックして、インポートが成功したことを確認します。**

```
# svcs mqbroker
```

出力は次のようになります。

```
STATE STIME FMRI
disabled 16:22:50 svc:/application/sun/mq/mqbroker:default
```

デフォルトでは、サービスは無効 (disabled) と示されます。

3 **mqbroker** サービスを有効にします。

```
# svcadm enable svc:/application/sun/mq/mqbroker:default
```

mqbroker サービスを有効にすると、`imqbrokerd` プロセスが開始されます。再起動すると、ブローカも再起動されます。

4 必要な引数を `imqbrokerd` コマンドに渡すように **mqbroker** サービスを設定します。

`imqbrokerd` に引数を渡すには、`options/server_args` プロパティーを使用します。たとえば、`-loglevel DEBUGHIGH` を追加するには、次のように指定します。

```
# svccfg
svc:> select svc:/application/sun/mq/mqbroker
svc:/application/sun/mq/mqbroker> setprop options/server_args=\"-loglevel DEBUGHIGH\"
svc:/application/sun/mq/mqbroker> exit
```

JMX API の変更点

Message Queue では、Message Queue クライアントアプリケーション内部からプログラムによってブローカ関数を設定および監視するための Java Management Extensions (JMX) API がサポートされます。Message Queue 4.2 には、このリリースの新機能をサポートするための、JMX API の拡張機能が含まれています。次の Mbean に対して、JMX の新しい属性、操作、検索キーのすべてまたはいずれかが定義されています。

- 34 ページの「コンシューママネージャーの監視 MBean」
- 35 ページの「送信先設定 MBean」
- 36 ページの「送信先マネージャーの設定 MBean」
- 36 ページの「送信先監視 MBean」
- 38 ページの「プロデューサマネージャーの監視 MBean」

コンシューママネージャーの監視 MBean

次の各表に示す属性、操作、および検索キーは、11 ページの「パブリッシャーまたはサブスクライバの複数の送信先」で説明した機能をサポートします。

次の属性の名前は、ユーティリティークラス `com.sun.messaging.jms.management.server.ConsumerAttributes` の静的定数として定義されています。

表 1–11　コンシューママネージャーの監視属性

名前	型	設定可/不可	説明
NumWildcardConsumers	整数	不可	ブローカに関連付けられているワイルドカードメッセージコンシューマの数

次の操作の名前は、ユーティリティークラス com.sun.messaging.jms.management.server.ConsumerOperations の静的定数として定義されています。

表 1–12　コンシューママネージャーの監視操作

名前	パラメータ	結果の型	説明
getConsumerWildcards	なし	String[]	ブローカに関連付けられている現在のコンシューマが使用するワイルドカード文字列
getNumWildcardConsumers	ワイルドカード文字列	整数	指定されたワイルドカード文字列を使用している、ブローカに関連付けられている現在のコンシューマの数

次の検索キーの名前は、ユーティリティークラス com.sun.messaging.jms.management.server.ConsumerInfo の静的定数として定義されています。

表 1–13　メッセージコンシューマ情報の検索キー

名前	値の型	説明
DestinationNames	String[]	ワイルドカードコンシューマが使用するワイルドカードと一致する送信先名 トピック送信先でのみ使用。
Wildcard	ブール型	ワイルドカードコンシューマかどうか トピック送信先でのみ使用。

送信先設定 MBean

次の表に示す属性は、13 ページの「XML ペイロードメッセージのスキーマ検証」で説明した機能をサポートします。

次の属性の名前は、ユーティリティークラス com.sun.messaging.jms.management.server.DestinationAttributes の静的定数として定義されています。

表 1–14　送信先設定属性

名前	型	設定可/不可	説明
ValidateXMLSchemaEnabled	ブール型	可	XML スキーマ検証が有効になっているかどうか false に設定されているか、または何も設定されていない場合、送信先の XML スキーマ検証は有効になっていません。
XMLSchemaURIList	文字列	可	XML スキーマドキュメント (XSD) URI 文字列のスペース区切りリスト URI は、XML スキーマ検証が有効になっている場合に使用する 1 つ以上の XSD の場所を示します。 複数の URI を指定する場合は、この値を二重引用符で囲みます。 例: “http://foo/flap.xsd http://test.com/test.xsd” このプロパティーが設定されていないか、または null の場合に XML 検証が有効になっていると、XML ドキュメント内で指定された DTD を使用して XML 検証が実行されます。
ReloadXMLSchemaOnFailure	ブール型	可	障害発生時に XML スキーマを再読み込みするかどうか false に設定されているか、または何も設定されていない場合は、検証で障害が発生したときにスキーマの再読み込みは行われません。

送信先マネージャーの設定 MBean

13 ページの「XML ペイロードメッセージのスキーマ検証」で説明した新機能をサポートする新しい送信先設定 MBean 属性は、送信先マネージャー設定 MBean の create 操作で送信先を作成するときに使用できます。

送信先監視 MBean

次の表に示す最初の属性セットは、11 ページの「パブリッシャーまたはサブスクライバの複数の送信先」で説明した機能をサポートし、2 番目の属性セットは、32 ページの「新しい送信先メトリック」で説明した拡張機能をサポートします。

次の属性の名前は、ユーティリティークラス
`com.sun.messaging.jms.management.server.DestinationAttributes` の静的定数として定義されています。

表 1–15　送信先監視属性

名前	型	設定可/不可	説明
`NumWildcards`	整数	不可	送信先に関連付けられているワイルドカードメッセージプロデューサとワイルドカードメッセージコンシューマの現在の数 トピック送信先でのみ使用。
`NumWildcardProducers`	整数	不可	送信先に関連付けられているワイルドカードメッセージプロデューサの現在の数 トピック送信先でのみ使用。
`NumWildcardConsumers`	整数	不可	送信先に関連付けられているワイルドカードメッセージコンシューマの現在の数 トピック送信先でのみ使用。
`NumMsgsRemote`	ロング整数	不可	メモリーおよび持続ストアに格納されている、クラスタ内のリモートブローカに対して生成されたメッセージの現在の数。この数には、トランザクションに含まれているメッセージは含まれません。
`TotalMsgBytesRemote`	ロング整数	不可	メモリーおよび持続ストアに格納されている、クラスタ内のリモートブローカに対して生成されたメッセージの合計サイズ (バイト単位)。この値には、トランザクションに含まれているメッセージは含まれません。

次の表に示す操作は、11 ページの「パブリッシャーまたはサブスクライバの複数の送信先」で説明した機能をサポートします。

次の操作の名前は、ユーティリティークラス
`com.sun.messaging.jms.management.server.DestinationOperations` の静的定数として定義されています。

表 1–16　送信先監視操作

名前	パラメータ	結果の型	説明
getWildcards	なし	String[]	送信先に関連付けられている現在のコンシューマおよびプロデューサが使用するワイルドカード文字列 トピック送信先でのみ使用。
getConsumerWildcards	なし	String[]	送信先に関連付けられている現在のコンシューマが使用するワイルドカード文字列 トピック送信先でのみ使用。
getProducerWildcards	なし	String[]	送信先に関連付けられている現在のプロデューサが使用するワイルドカード文字列 トピック送信先でのみ使用。
getNumWildcardConsumers	ワイルドカード文字列	整数	指定されたワイルドカード文字列を使用している、送信先に関連付けられている現在のコンシューマの数 トピック送信先でのみ使用。
getNumWildcardProducers	ワイルドカード文字列	整数	指定されたワイルドカード文字列を使用している、送信先に関連付けられている現在のプロデューサの数 トピック送信先でのみ使用。

プロデューサマネージャーの監視 MBean

次の各表に示す属性、操作、および検索キーは、11 ページの「パブリッシャーまたはサブスクライバの複数の送信先」で説明した機能をサポートします。

次の属性の名前は、ユーティリティークラス `com.sun.messaging.jms.management.server.ProducerAttributes` の静的定数として定義されています。

表 1–17　プロデューサマネージャーの監視属性

名前	型	設定可/不可	説明
NumWildcardProducers	整数	不可	ブローカに関連付けられているワイルドカードメッセージプロデューサの数

次の操作の名前は、ユーティリティークラス `com.sun.messaging.jms.management.server.ProducerOperations` の静的定数として定義されています。

表 1–18　プロデューサマネージャーの監視操作

名前	パラメータ	結果の型	説明
getProducerWildcards	なし	String[]	ブローカに関連付けられている現在のプロデューサが使用するワイルドカード文字列
getNumWildcardProducers	ワイルドカード文字列	整数	指定されたワイルドカード文字列を使用している、ブローカに関連付けられている現在のプロデューサの数

次の検索キーの名前は、ユーティリティークラス com.sun.messaging.jms.management.server.ProducerInfo の静的定数として定義されています。

表 1–19　メッセージプロデューサ情報の検索キー

名前	値の型	説明
DestinationNames	String[]	ワイルドカードプロデューサが使用するワイルドカードと一致する送信先名 トピック送信先でのみ使用。
Wildcard	ブール型	ワイルドカードプロデューサかどうか トピック送信先でのみ使用。

クライアント認証での DN ユーザー名形式のサポート

Message Queue 4.2 では、LDAP ユーザーリポジトリによるクライアント接続認証で DN ユーザー名形式がサポートされます。そのため、次の新しいブローカプロパティー (および値) が追加されています。

```
imq.user_repository.ldap.usrformat=dn
```

このプロパティーにより、ブローカは、次のプロパティーで指定された属性の値を DN ユーザー名形式から抽出することにより、LDAP ユーザーリポジトリのエントリと照合してクライアントユーザーを認証します。

```
imq.user_repository.ldap.uidattr
```

ブローカは、アクセス制御操作で、前述の属性の値をユーザー名として使用します。

たとえば、imq.user_repository.ldap.uidattr=udi と指定したときに、クライアント認証ユーザー名の形式が udi=mquser,ou=People,dc=red,dc=sun,dc=com の場合、アクセス制御の実行時に「mquser」が抽出されます。

JAAS 認証の拡張機能

Message Queue 4.2 の JAAS 認証では、ユーザー名による認証に加えて、IP アドレスによる認証もサポートされます。

既知の問題点と制限事項

この節には、Message Queue 4.2 の既知の問題についてのリストが含まれています。次の内容について説明します。

現時点のバグ、その状態、および回避策の一覧については、Java Developer Connection™ メンバーは、Java Developer Connection Web サイトの「Bug Parade」ページを参照してください。新しいバグを報告する前に、このページをチェックしてください。すべての Message Queue バグがリストされているわけではありませんが、このページはある問題が報告済みかどうかを知りたい場合に活用できます。

```
http://bugs.sun.com/bugdatabase/index.jsp
```

注 – Java Developer Connection のメンバーになるのは無料ですが、登録が必要です。Java Developer Connection のメンバーになる方法についての詳細は、Sun の「For Developers」Web ページを参照してください。

新しいバグの報告や機能に関する要求を行うには、`imq-feedback@sun.com` 宛てにメールを送信してください。

インストールに関する情報

この節では、Message Queue version 4.2 のインストールに関連した問題について説明します。

製品レジストリと Java ES

Message Queue 4.2 は、Message Queue 4.1 と同様に、新しいインストーラでインストールされます。このインストーラでは、JDK、NSS、JavaHelp など、Message Queue に必要な Java Enterprise System (Java ES) 共有コンポーネントもインストールされます。

新しい Message Queue インストーラと、以前のバージョンの Message Queue で使用されていた古い Java ES インストーラは、同じ製品レジストリを共有しません。Java ES でインストールされたあるバージョンの Message Queue が削除され、Message Queue インストーラで Message Queue 4.2 にアップグレードされた場合、Java ES 製品レジストリは矛盾する状態になることがあります。その結果、Java ES アンインストーラを実行すると、Java ES でインストールしなかったとしても、Message Queue 4.2 とそれが依存する共有コンポーネントが意図せずに削除されることがあります。

Java ES インストーラでインストールした Message Queue ソフトウェアをアップグレードする最善の方法は、次のとおりです。

1. Java ES アンインストーラを使用して、Message Queue とその共有コンポーネントを削除します。
2. Message Queue インストーラを使用して、Message Queue 4.2 をインストールします。

Windows でのインストール

Message Queue を Windows にインストールするときは、次の制限事項に注意してください。

- インストーラは、Message Queue 用のエントリを「スタート」>「すべてのプログラム」メニューに追加しません。(バグ *6567258*)

  回避方法*:* 管理コンソールを起動するには、『Sun Java System Message Queue 4.2 Administration Guide』の「Starting the Administration Console」に示されているように、コマンド行を使用します。
- インストーラは、`IMQ_HOME\mq\bin` ディレクトリを `PATH` 環境変数に追加しません。*(バグ 6567197)*

  回避方法*:* ユーザーは Message Queue ユーティリティー (`IMQ_HOME\mq\bin\`*command*) を起動するときに、このエントリを `PATH` 環境変数に追加するか、フルパス名を指定する必要があります。
- インストーラは、Message Queue がインストールされたことを示すために Windows レジストリにエントリを追加しません。*(バグ 6586389)*
- 回答ファイルを使用してサイレントモードで実行すると、インストーラはすぐに返されます。インストールは行われますが、ユーザーにはサイレントインストールがいつ実際に実行されたかを知る方法がありません。 *(バグ 6586560)*
- Windows で、インストーラをテキストモード (`installer –t`) で実行しようとすると、エラーメッセージが表示されます。このメッセージは、インストーラが英語以外で実行されていても、英語で表示されます。テキストモードは Windows ではサポートされていません。*(バグ 6594142)*
- インストーラは、デフォルトでは、オペレーティングシステムがインストールされているドライブに Message Queue をインストールしません。*(バグ 6673511)*

- Windows でのインストールおよびアンインストールでは、ユーザーが実行できる bat ファイルはありません。また、ユーザーが Windows コントロールパネルの「プログラムの追加と削除」を使用してアンインストールすることもできません。*(バグ 6673417)*
- Windows Vista では、管理者としてコマンドプロンプトからインストールしないかぎり、Message Queue を C:\Program Files の下にインストールできません。*(バグ 6701661)*

  回避方法: 管理者としてコマンドプロンプトからインストールするには、次の手順に従います。

  1.「スタート」 →「すべてのプログラム」 →「アクセサリ」の順にクリックし、「コマンド プロンプト」をポイントします。

  2.「コマンド プロンプト」を右クリックします。

  3.「管理者として実行」をクリックします。

  4. ディレクトリを Message Queue 4.2 のインストールイメージに変更します。

  5. `installer.vbs` を実行します。
- アンインストーラを予行モード (`uninstaller -n`) で実行すると、アンインストールが誤って実行されます。*(バグ 6719051)*

  回避方法: 次のコマンドを使用して、サイレントインストールを実行します。

  ```
  uninstaller -s
  ```
- インストーラのホームページで「Install Home」の文字列がローカライズされません。*(バグ 6592491)*

Solaris でのインストール

- インストーラを予行モード (`installer –n`) で実行すると、「概要」画面にいくつかのエラーメッセージが表示され、「未完了」のインストール状態も表示されます。これは誤りで、誤解を招きます。予行モードでは、何もシステムにインストールされません。後でサイレントインストールの実行に使用できる回答ファイルが作成されるだけです。*(バグ 6594351)*
- 回答ファイルを使用してサイレントモード (`installer -a` *filename* `-s`) でインストーラを実行すると、Sun Connection への登録が実行されません。(バグ 6710268)
- テキストモードでインストーラを実行している場合、Sun Connect 登録用のユーザー名またはパスワードを入力したり、オンラインアカウントを作成したりするときに、ユーザー名やパスワードの訂正にバックスペースキーを使用できません。*(バグ 6673460)*

  回避方法: バックスペースキーの代わりに `Control-H` キーを使用するか、または `dtterm` や `xterm` のような別の端末エミュレータを使用します。

- インストーラの「アップグレード」画面に、インストールされている既存の Message Queue またはインストーラエンジンのバージョが常に正しく表示されません。(バグ 6679765)
- テキストモードでインストーラを実行しているときに、無効なユーザー名とパスワードで Sun Connection に登録しようとすると、登録できないことを示すメッセージが表示され、Null ポインタ例外がスローされて、インストーラが終了します。(バグ 6666365)

Linux でのインストール

次に示す問題は、Linux プラットフォームでのインストールに影響します。

- 「JDK 選択」パネルで、スクロールリストに項目が 1 つしか表示されません。このため、リスト内のほかの JDK を選択することが難しくなります。*(バグ 6584735)*
- JDK が現在のもので、ユーザーが「JDK 選択」画面で「Java(TM) SDK のデフォルトバージョンをインストールして使用します。」を選択した場合、インストーラはその JDK をインストールしようとし、パッケージをインストールできないというレポートを返します。この問題があっても、インストールは正常に完了します。*(バグ 6581310)*
- 現在インストールされている JDK が JDK 1.5.0_15 (Message Queue インストーラによって標準でインストールされるバージョン) より新しいバージョンの場合、Message Queue アンインストーラはデフォルトの IMQ_JAVAHOME ディレクトリを見つけることができず、エラーを返します。*(バグ 6673415)*

  回避方法: Message Queue アンインストーラを実行する前に、次のコマンドを使用して JDK 1.5 を手動でインストールします。

  ```
  # cd installImage/Product/UNIX/LINUX/X86/2.4/Packages
  # rpm -i --force jdk-1.5.0_15–linux- arch.rpm
  ```

  arch は `i586` か `amd64` のいずれかです。
- インストーラを予行モード (`installer –n`) で実行すると、「概要」画面にいくつかのエラーメッセージが表示され、「未完了」のインストール状態も表示されます。これは誤りで、誤解を招きます。予行モードでは、何もシステムにインストールされません。後でサイレントインストールの実行に使用できる回答ファイルが作成されるだけです。*(バグ 6594351)*

すべてのプラットフォームでのインストール

次に示す問題は、すべてのプラットフォームでのインストールに影響します。

- 「インストールの準備完了」画面で、製品名が「Sun Java System Message Queue 4.2」ではなく「mq」と表示されます。*(バグ 6650841)*

- インストーラが Message Queue 4.2 のインストールを処理中で、「進行状況」画面が表示されているときは、「取消し」ボタンがアクティブになります。ここで「取消し」ボタンを選択すると、インストールが完了しないか、破棄されます。*(バグ 6595578)*
- インストーラの「概要」画面には、クリックするとログまたは概要ページビューアが開くリンクがいくつか含まれています。「閉じる」というラベルの付いたボタンの代わりに、ウィンドウの閉じるボタン「X」を使用してこのビューアウィンドウを取り消した場合は、このビューアウィンドウを再度表示することはできません。*(バグ 6587138)*

  回避方法:「閉じる」というラベルの付いたボタンを使用してウィンドウを閉じます。
- コンピュータシステムに古いバージョンの Message Queue および NSS/NSPR があると、インストーラの「アップグレード」画面で、アップグレードが必要なものとして Message Queue だけが表示され、NSS と NSPR のアップグレードも必要であることは触れられません。実際には、関連ソフトウェアもすべてアップグレードされ、「インストールの準備完了」画面では正しい情報が示されます。*(バグ 6580696)*
- 「JDK の選択」画面の JDK のリストが、「JDK を選択」オプションが選択されていない場合でもアクティブになります。*(バグ 6650874)*

インストーラのバージョン表示の問題

インストーラには、Message Queue のバージョン情報が不明瞭な形式で表示されます。*(バグ 6586507)*

Solaris プラットフォームの場合は、次の表を参照して、インストーラに表示される Message Queue のバージョンを判別してください。

表 **1–20** バージョン文字列の読み換え

Solaris OS でインストーラに表示されるバージョン	対応する **Message Queue** のリリース
4.2.0.0	4.2
4.1.0.2	4.1 Patch 2
4.1.0.1	4.1 Patch 1
4.1.0.0	4.1
3.7.2.1	3.7 UR2 Patch 1
3.7.0.2	3.7 UR2
3.7.0.1	3.7 UR1

表 1–20 バージョン文字列の読み換え (続き)

Solaris OS でインストーラに表示されるバージョン	対応する Message Queue のリリース
3.6.0.0	3.6
3.6.0.4	3.6 SP4
3.6.0.3	3.6 SP3
3.6.0.2	3.6 SP2
3.6.0.1	3.6 SP1

注 – 3.6 SP4 のパッチリリースの場合 (たとえば、3.6 SP4 Patch 1)、インストーラで表示されるリリース文字列は同じままです。厳密なバージョンを判別するには、コマンド `imqbrokerd -version` を実行する必要があります。

Linux のプラットフォームの場合、インストーラで表示されるバージョン番号は次の形式になります。

majorReleaseNumber.*minorReleaseNumber*-*someNumber*

たとえば、3.7–22 のようになります。これにより、3.7 リリースの 1 つであることだけはわかりますが、どの特定のバージョンかはわかりません。インストールされている Message Queue のバージョンを判別するには、次のコマンドを実行します。

```
imqbrokerd -version.
```

ローカリゼーションの問題

次に示す問題は、ローカリゼーションの問題に関係しています。

- インストーラを英語以外のロケールでテキストモード (`installer –t`) で実行すると、マルチバイト文字が文字化けします。 *(バグ 6586923)*
- インストーラの「進行状況」画面で、進行状況のバーに見慣れない文字が表示されます。ツールヒントは、英語以外のロケールではハードコードされています。 *(バグ 6591632)*
- テキストモード (`installer –t`) は Windows ではサポートされていません。Windows でテキストモードでインストーラを実行すると、エラーメッセージが表示されます。このメッセージは、インストーラが英語以外のロケールで実行されていても、ローカライズされません。*(バグ 6594142)*
- インストーラの「ライセンス」画面には、インストーラが実行されるロケールに関係なく、英語のライセンステキストが表示されます。*(バグ 6592399)*

  回避方法: ローカライズされたライセンスファイルにアクセスするには、`LICENSE_MULTILANGUAGE.pdf` ファイルを検索してください。

- インストーラの使用方法に関するヘルプテキストがローカライズされていません。*(バグ 6592493)*
- インストーラの概要 HTML ページに表示される文字列「None」は、英語でハードコードされています。*(バグ 6593089)*
- インストーラをドイツ語のロケールで実行すると、ほかのロケールでは表示される開始画面のテキストが正しく表示されません。*(バグ 6592666)*
- インストーラの「Install Home」画面に表示される文字列「Install Home」がローカライズされていません。その文字列は、英語以外のロケールでインストーラを実行しても、英語で表示されます。*(バグ 6592491)*
- テキストモード (`installer –t`) でインストーラを実行すると、どのロケールでインストーラを実行するかに関係なく、英語の応答選択肢「Yes」および「No」が使用されます。*(バグ 6593230)*
- インストーラの「JDK 選択」画面の参照ボタンのツールヒントは、英語でハードコードされています。*(バグ 6593085)*

使用されなくなったパスワードオプション

以前のバージョンの Message Queue では、`–p` または `–password` オプションを使用して、次のようなコマンドのパスワードを対話形式で指定することができました。`imqcmd`、`imqbrokerd`、`imdbmgr` version 4.0 から、これらのオプションは使用できなくなりました。

代わりに、関連するパスワードを指定したパスワードファイルを作成し、`-passfile` コマンドオプションを使用してそのパスワードファイルを参照できます。または、コマンドで要求されたときにパスワードを入力することもできます。

パスワードファイルには、次に示すパスワードを 1 つ以上格納することができます。

- SSL キーストアを開くために使用するキーストアパスワード。このパスワードを指定するには `imq.keystore.password` プロパティーを使用します。
- 接続が匿名でない場合に LDAP ディレクトリとの接続のセキュリティーを確保するために使用する LDAP リポジトリパスワード。このパスワードを指定するには `imq.user_repository.ldap.password` プロパティーを使用します。
- JDBC 準拠のデータベースとの接続に使用する JDBC データベースパスワード。このパスワードを指定するには `imq.persist.jdbc.vendorName.password` プロパティーを使用します。プロパティー名の *vendorName* コンポーネントは、データベースベンダーを指定する変数です。`hadb`、`derby`、`pointbase`、`oracle`、`mysql` から選択できます。
- `imqcmd` コマンド (ブローカ管理タスクを実行する) に対するパスワード。このパスワードを指定するには `imq.imqcmd.password` プロパティーを使用します。

次の例では、JDBC データベースに対するパスワード abracadabra をパスワードファイルに設定しています。

```
imq.persist.jdbc.mysql.password=abracadabra
```

パスワードファイルは、次のいずれかの方法で使用できます。

- ブローカがパスワードファイルを使用するように、ブローカの config.properties ファイルに次のプロパティーを設定する。

  ```
  imq.passfile.enabled=true
  imq.passfile.dirpath=passwordFileDirectory
  imq.passfile.name=passwordFileName
  ```

- 関連するコマンドの -passfile オプションを使用する。次に例を示します。

  ```
  imqbrokerd -passfile passwordFileName
  ```

管理および設定上の問題

次に示す問題は Message Queue の管理および設定に関係するものです。

- Windows プラットフォームの場合、デフォルトで有効になっている組み込みの Windows ファイアウォールで、ブローカがクライアントから受信接続を受け付けるようにするためのファイアウォール規則を手動で設定する必要があります。(バグ *6675595*)

  1. コントロールパネルの「Windows ファイアウォール」をダブルクリックします。

     Windows ファイアウォールの設定ダイアログを開くには、「ユーザー アカウント制御」ダイアログで「続行」をクリックします。

  2. Windows ファイアウォールの設定ダイアログで、「例外」タブをクリックします。
  3. 「プログラムの追加」をクリックします。
  4. 「プログラムの追加」ダイアログで、「java.exe」を選択し、「参照」をクリックします。

     Windows はブローカプロセスを Java Platform SE バイナリとして識別します。従って、ブローカが使用する java.exe (通常は jdk1.5.0_15\jre\bin\java.exe) を探します。

  5. 「スコープの変更」をクリックします。
  6. 「スコープの変更」ダイアログで、「任意のコンピュータ (インターネット上のコンピュータを含む)」を選択します。
  7. 「OK」をクリックします。
  8. 「プログラムの追加」ダイアログで「OK」をクリックします。

9. Windows ファイアウォールの設定ダイアログで「OK」をクリックします。

- Windows プラットフォームの場合、`CLASSPATH` に二重引用符が含まれていると、`imqadmin` および `imqobjmgr` コマンドの実行時にエラーがスローされます。(バグ *5060769*)

  回避方法: コマンドプロンプトウィンドウを開き、次のように入力して `CLASSPATH` を設定解除します。

  ```
  set classpath=
  ```

  その後、同じコマンドプロンプトウィンドウで、必要なコマンドを実行します。次に例を示します。

  mqInstallHome`\mq\bin\imqadmin`

- すべての Solaris および Windows スクリプトで、`-javahome` オプションの値に空白文字が含まれると動作しません。(バグ *4683029*)

  `javahome` オプションは Message Queue コマンドおよびユーティリティーで使用し、使用する代替の Java 2 互換のランタイムを指定します。ただし、代替の Java 2 互換のランタイムへのパスには、空白文字を含めることはできません。空白文字を含むパスの例は次のとおりです。

  Windows の場合: `C:\jdk 1.4`

  Solaris の場合: `/work/java 1.4`

  回避方法: Java ランタイムを、空白文字が含まれない場所またはパスにインストールします。

- `imqQueueBrowserMaxMessagesPerRetrieve` 属性は、クライアントランタイムがキューの内容を検索するときに一度に取得することのできるメッセージの最大数を指定します。この属性は、キューに入れられたメッセージがバッチ処理され、クライアントランタイムに配信される方法には影響しますが、参照されるメッセージの総数には影響しません。この属性は参照機構のみに影響し、キューのメッセージの配信には影響しません。*(バグ 6387631)*

ブローカの問題

次に示す問題は Message Queue ブローカに影響します。

- 持続データストアがあまりにも多くの送信先を開く場合、ブローカがアクセス不可能になります。(バグ *4953354*)

  回避方法: この状態はブローカがシステムのオープンファイル記述子の制限に達したことが原因です。Solaris や Linux では、`ulimit` コマンドを使って、ファイル記述子の制限を増やします。

- 送信先が破棄された場合、コンシューマが孤立します。(バグ *5060787*)

送信先が破棄された場合、アクティブコンシューマが孤立します。いったんコンシューマが孤立すると、送信先が再作成された場合でもメッセージを受信しなくなります。

- HTTP 接続サービスを使用している JMS クライアントが、`Ctrl-C` の使用などにより突然終了した場合、ブローカがクライアント接続や関連するすべてのリソースを解放するまでに、およそ 1 分かかります。

  この 1 分の間にクライアントのほかのインスタンスが起動し、同じ ClientID、永続サブスクリプション、またはキューを使おうとした場合、そのインスタンスは「クライアント ID はすでに使用されています」の例外を受け取ります。このことは実際の問題ではなく、上記の終了処理の結果にすぎません。およそ 1 分経過後にクライアントが起動すると、すべて問題なく動作します。

- データストアに MySQL データベースを使用している場合、1M バイトを超えるメッセージを格納すると、「クエリーのパケットが大きすぎます...」という SQLException がスローされます。*(バグ 6682815)*

  回避方法: `--max_allowed_packet` オプションでデフォルトの 1M バイトより大きい値を設定して MySQL サーバーを起動します。たとえば、次の値を使用します。

  ```
  --max_allowed_packet=60M
  ```

- データストアに Java DB データベースを使用している場合、メッセージを格納すると、「要求された時間内に例外を取得できませんでした」という SQLException がスローされます。*(バグ 6691394)*

  回避方法: ブローカの `config.properties` ファイルに次のプロパティー値を設定します。:

  ```
  imq.persist.jdbc.derby.table.MYCONSTATE41.index.IDX2=CREATE INDEX &(index)
  ON $(name) (MESSAAGE_ID)
  ```

- 高可用性共有データストアに MySQL データベースを使用している場合、MySQL ストレージエンジンを `NDBCLUSTER` として設定する機構が必要です。*(バグ 6691394)*

  回避方法: ブローカの `config.properties` ファイルに次のプロパティー値を設定します。:

  ```
  imq.persist.jdbc.mysql.tableoption=EMGINE=NDBCLUSTER
  ```

ブローカクラスタ

次に示す問題は、ブローカクラスタに影響します。

- このリリースでは、フル接続のブローカクラスタのみサポートされています。つまり、クラスタ内のすべてのブローカは、そのクラスタ内のほかのブローカと相互に直接やり取りする必要があります。`imqbrokerd -cluster` コマンド行引数を使用してブローカを従来のクラスタに接続する場合は、そのクラスタ内のすべてのブローカが含まれていることを確認してください。

- クライアントが高可用性ブローカクラスタ内のブローカに接続されている場合、クライアントランタイムは、`imqAddressListIterations` 接続ファクトリ属性の値を無視して、成功するまで再接続を試みます。
- クライアントが検索できるのは、ホームブローカにあるキューの内容のみです。この場合でも、クライアントは、クラスタ内の任意のブローカに対してキューにメッセージを送信したりキューからのメッセージを消費したりできます。制約を受けるのはキューの検索のみです。
- Version 4.2 のブローカが含まれる従来のクラスタでは、すべてのクラスタが Version 3.5 以降でなければなりません。
- デフォルトでは、Message Queue 4.2 および 4.1 のブローカを Message Queue 3.7 または 3.6 のブローカと同時に使用できません。これは、これらのバージョン間で `imq.autocreate.queue.maxNumActiveConsumers` のデフォルト値が変更されているためです。*(バグ 6716400)*

  回避方法: Message Queue 4.2 および 4.1 のブローカの `imq.autocreate.queue.maxNumActiveConsumers` の値を、デフォルト値の `-1` から以前のバージョンのデフォルト値である `1` に変更します。
- 従来のクラスタから高可用性クラスタに変換するときに、Message Queue データベースマネージャーユーティリティー (`imqdbmgr`) を使用して、既存のスタンドアロン JDBC ベースデータストアから高可用性共有データストアに変換できます。方法については、『Sun Java System Message Queue 4.2 Administration Guide』の「Converting a Standalone Data Store to a Shared Data Store」を参照してください。
- HADB を使用するブローカは、10M バイトより大きなメッセージを処理できません。*(バグ 6531734)*
- コマンド `imqdbmgr upgrade hastore` を使用した HADB ストアへの変換は、ストアに 10,000 を超えるメッセージが保持されている場合、「設定されているロックの数が多すぎます」というメッセージが表示されて失敗することがあります。*(バグ 6588856)*

  回避方法: 次のコマンドを使用して、ロックの数を増やします。

  ```
  hadbm set NumberOfLocks=<desiredNumber>
  ```

  詳細は、『Sun Java System Application Server 9.1 Enterprise Edition トラブルシューティングガイド』の「HADB の問題」を参照してください。
- 1 つのトランザクションに 500 を超えるリモートメッセージがコミットされている場合、ブローカはエラー「HADB-E-12815: Table memory space exhausted」を返すことがあります。*(バグ 6550483)*

  詳細は、『Sun Java System Application Server 9.1 Enterprise Edition トラブルシューティングガイド』の「HADB の問題」を参照してください。
- ブローカクラスタで、開始していないリモート接続へのメッセージをブローカがキューに入れます。(バグ *4951010*)

回避方法: いったんその接続が開始すると、メッセージはコンシューマによって受信されます。コンシューマの接続が閉じている場合、メッセージは別のコンシューマへ再配信されます。

- 1 つのトランザクションでリモートブローカから複数のメッセージを消費しているときは、次のエラーメッセージがブローカに記録される可能性があります。このメッセージは無視しても問題ありません。

```
[26/Jul/2007:13:18:27 PDT] WARNING [B2117]:
Message acknowledgement failed from
mq://129.145.130.95:7677/?instName=a&brokerSessionUID=3209681167602264320:
    ackStatus = NOT_FOUND(404)\
    Reason = Update remote transaction state to COMMITED(6):
transaction 3534784765719091968 not found, the transaction
may have already been committed.
    AckType = MSG_CONSUMED
    MessageBrokerSession = 3209681167602264320
    TransactionID = 3534784765719091968
        SysMessageID = 8-129.145.130.95(95:fd:93:91:ec:a0)-33220-1185481094690
        ConsumerUID = 3534784765719133952\par

[26/Jul/2007:13:18:27 PDT] WARNING Notify commit transaction
[8-129.145.130.95(95:fd:93:91:ec:a0)-33220-1185481094690,
[consumer:3534784765719133952, type=NONE]]
TUID=3534784765719091968 got response:
com.sun.messaging.jmq.jmsserver.util.BrokerException:
 Update remote transaction state to COMMITED(6):
  transaction 3534784765719091968 not found, the transaction may have already
  been committed.:
com.sun.messaging.jmq.jmsserver.util.BrokerException: Update remote transaction
state to COMMITED(6): transaction 3534784765719091968 not found, the transaction
 may have already been committed.r
```

このメッセージは、`imq.txn.reapLimit` プロパティーが 1 つのトランザクション内のリモートメッセージの数と比較して少ない場合に、トランザクション内の後のメッセージのためにコミットをメッセージホームブローカに通知するときに記録されます。*(バグ 6585449)*

回避方法: このメッセージを回避するには、`imq.txn.reapLimit` プロパティーの値を増やします。

JMX の問題

Windows プラットフォームでは、トランザクションマネージャーの監視 MBean の `getTransactionInfo` メソッドが不正なトランザクションの作成時間を含むトランザクション情報を返します。(バグ *6393359*)

回避方法: 代わりにトランザクションマネージャーの監視 MBean の `getTransactionInfoByID` メソッドを使用します。

SOAP サポート

SOAP サポートに関連した 2 つの問題に注意する必要があります。

- Message Queue の version 4.0 のリリースから、SOAP 管理によるオブジェクトのサポートは中止されています。
- SOAP 開発は、いくつかのファイルに依存しています。`SUNWjaf`、`SUNWjmail`、`SUNWxsrt`、および `SUNWjaxp` です。Message Queue の version 4.1 では、これらのファイルを利用できるのは JDK version 1.6.0 以降で Message Queue を実行している場合のみです。
- 以前は、SAAJ 1.2 実装 .jar が `mail.jar` を直接参照していました。SAAJ 1.3 では、この参照が削除されたため、Message Queue クライアントが `mail.jar` を明示的に `CLASSPATH` 内に配置する必要があります。

再配布可能なファイル

Sun Java System Message Queue 4.2 には、バイナリ形式で使用でき自由に配布可能な次のファイルのセットが含まれています。

`fscontext.jar`
`imq.jar`
`imqjmx.jar`
`imqxm.jar`
`jaas.jar`
`jms.jar`
`libmqcrt.so` (HPUX)
`libmqcrt.so` (UNIX)
`mqcrt1.dll` (Windows)

さらに、`LICENSE` ファイルおよび `COPYRIGHT` ファイルも再配布できます。

障害を持つユーザー向けのアクセシビリティー機能

このメディアの出版後にリリースされたアクセシビリティーを入手する場合は、請求に応じて Sun から提供される 508 条に関する製品評価資料を参照し、使いやすいソリューションの配備にもっとも適したバージョンを調べてください。最新バージョンのアプリケーション
は、`http://sun.com/software/javaenterprisesystem/get.html` にあります。

アクセシビリティーに対する Sun の取り組みについては、`http://sun.com/access` を参照してください。

問題の報告とフィードバックの方法

Sun Java System Message Queue に問題が発生した場合は、次のいずれかの方法で Sun のカスタマサポートにお問い合わせください。

- Sun ソフトウェアサポートサービス http://www.sun.com/service/sunone/software

  このサイトには、ナレッジベース、オンラインサポートセンター、ProductTracker へのリンクと保守プログラムおよびサポートの連絡先電話番号へのリンクがあります。
- 保守契約を結んでいるお客様の場合は、専用ダイヤルをご利用ください。

最善の問題解決のため、サポートに連絡する際には次の情報をご用意ください。

- 問題が発生した箇所や動作への影響など、問題の具体的な説明
- マシンのタイプ、OS のバージョン、および製品のバージョン (問題に関連している可能性のあるパッチやその他のソフトウェアを含む)
- 問題を再現するために用いた詳細な手順。
- すべてのエラーログまたはコアダンプ。

Sun Java System Software Forum

次のサイトでは、Sun Java System Message Queue フォーラムが利用できます。

http://swforum.sun.com/jive/forum.jspa?forumID=24

ご参加を歓迎いたします。

Java Technology Forum

Java Technology Forums には、関連する JMS のフォーラムがあります。

http://forum.java.sun.com

このマニュアルに関するコメント

弊社では、マニュアルの改善に努めており、お客様からのコメントおよびご忠告をお受けしております。

コメントを送るには、http://docs.sun.com にアクセスして「コメントの送信」をクリックしてください。このオンラインフォームでは、マニュアルのタイトルと Part No. もご記入ください。Part No. は、マニュアルのタイトルページか先頭に記述されている 7 桁または 9 桁の番号です。このマニュアルのタイトルは『Sun Java System Message Queue 4.2 リリースノート』で、Part No. は 820-5641 です。

その他の情報

次のインターネットのサイトで、Sun Java System の情報を参照できます。

- 関連文書

  http://docs.sun.com/prod/java.sys
- プロフェッショナルサービス

  http://www.sun.com/service/sunps/sunone
- ソフトウェア製品およびサービス

  http://www.sun.com/software
- ソフトウェアサポートサービス

  http://www.sun.com/service/sunone/software
- サポートおよびナレッジベース

  http://www.sun.com/service/support/software
- Sun サポートおよびトレーニングサービス

  http://training.sun.com
- コンサルティングおよびプロフェッショナルサービス

  http://www.sun.com/service/sunps/sunone
- 開発者向けの情報

  http://developers.sun.com
- Sun 開発者サポートサービス

  http://www.sun.com/developers/support
- ソフトウェアトレーニング

  http://www.sun.com/software/training